IN THE UNITED STATES PATENT
AND TRADEMARK OFFICE

Request for Rehearing
Appeal No. 2008-4652
Box Appeal Brief Patents
Honorable Commissioner of Patents and Trademarks
P.O. Box 1450
Alexandria, VA 22313-1450

Via Express Mail
Express Mail No. EH 692490026

Merged Proceedings:
Reissue Application No.:
09/512,592
United States Patent No.:
5,806,063
Issued: September 8, 1998
Applicant:
Dickens, Bruce M.
Reexamination Proceeding:
90/005,592
Filed: December 21, 1999
Reexamination Proceeding:
90/005,628
Filed: February 2, 2000
Reexamination Proceeding:
90/005,727
Filed: May 16, 2000
Reexamination Proceeding:
90/006,541
Filed: February 7, 2003

Group Art Unit: 2161

Examiner: LeRouz, E

APPLICANT'S REQUEST FOR REHEARING

Responsive to the Decision of the USPTO Board of Appeals and Interferences, applicant respectfully submits this Request for Rehearing under 37 C.F.R. §41.52, and requests the Board to consider the following points regarding the referenced issues ruled upon by the Board in the referenced Opinion portions in regard to the above captioned Merged Proceedings with respect to United States Patent No. 5806063 ("the Dickens patent").

Table of Contents

The Exhibit A Issue (Opinion, pp. 58-62)

The Board has indicated that Exhibit A is not part of the issued Dickens patent because applicant has so far not taken the right procedural steps to so make it a part. Regardless of the outcome of this request for Rehearing, applicant respectfully requests that the Board remand the application to the Examiner to allow applicant to proceed to have Exhibit A made part of the issued patent by the appropriate procedure. The Office has never taken the position that Exhibit A was not filed with the application leading to the Dickens patent, though not made part of the patent as issued.

The §112 Rejection of Claims 16-25, 26-30, 31-33, 66, 67, and 72 Based upon the Indefiniteness of the Term "Collectively" As Used in the Context of the Claimed Subject Matter (Opinion, pp. 67-72)

The Board has affirmed the rejections of claims 16-25, 26-30, 31-33, 66, 67, and 72 because of the indefiniteness of the term "collectively" as used in the claims. Applicant agrees with the Board that the common meaning for "collectively" is "formed by collecting: gathered into a mass, sum, or body: AGGREGATED" or "formed by collecting, gathered into a whole." The Board has taken the position that applicant:

> has not presented any convincing evidence that demonstrates that, as of the filing date of the '574 patent the inventor had possession of the claimed subject matter, i.e., 'collectively' performing the acts of 'further processing,' 'sorting,' 'manipulating,' or 'running a program.'

Applicant respectfully disagrees. The Specification and claims as originally filed, even without Exhibit A related to the claimed subject matter including "determining a century designator for *each* symbolic representation of a date in the database" followed by "reformatting the symbolic representation of the date [for each such symbolic representation of a date in the database] ... to facilitate further processing of the dates." This, along with the rest of the specification and claims clearly contemplates that the reformatted symbolic representations of each symbolic representation of a date in the database have been gathered into a mass, sum or

body or a whole, i.e., aggregated to facilitate further processing in the aggregated form, i.e., for being "collectively" further processed.[1]

The Board elsewhere in the opinion seems to have agreed with this interpretation of what the specification supports and the meaning of the claims to support "collectively" further processing.

The Board has agreed with applicant that the Dickens patent specification supports the recitation of, e.g., claim 33 "reformatting the symbolic representation of each symbolic representation of a date in the database, *without changing* any of the symbolic representations of a date in the database during the reformatting step." (Opinion, pp. 63-65) This clearly contemplates that the reformatted symbolic representations are aggregated after the reformatting step, or else how could they then be used for the remaining recited "facilitat[ion of] further processing."

The Board has agreed with applicant that the Dickens patent specification supports the recitation of, e.g., claim 20 "reformatting each symbolic representation of a date ... *separately* from the symbolic representations in the database." (Opinion, pp. 66-67) Similarly claim 62 recites "storing the converted symbolic representations separate from the at least one field of the database." If not the first of the noted recitations, then certainly the second involves aggregating the reformatted/converted symbolic representations, followed by further processing, expressly "running a program on the stored converted symbolic representations."

The Board has agreed with applicant that the Dickens patent specification supports the recitation of, e.g., claim 36 "sorting the converted symbolic representations *prior to* the step of *running a program on the converted symbolic representations*." (Opinion, p. 73-74) Sorting and then running a program on the sorted converted date data symbolic representations necessarily includes having aggregated the converted symbolic representations to run the program on them.

The Board has agreed with applicant that the Dickens patent specification supports the recitation of, e.g., claim 46, "converting *at least a substantial portion* of each of the plurality of symbolic representations of dates in at least one date field ... ." (Opinion, pp, 74-75) This recitation, especially in the context of the claim from which it depends (claim 34, "*running a program ... on each of the converted symbolic representations ... separately* from the date data

[1] The prosecution of the original application leading to the Dickens patent, as discussed in some detail in the December 22, 2002 Response to the Office Action of July 22, 2002, at pp. 26-29, supports the fact that original Examiner so interpreted the claims in allowing the claims in the originally issued Dickens patent.

symbolic representations contained in the ... database") not only distinguishes over Shaughnessy, but requires that the converted symbolic representations of at least a substantial portion of the symbolic representations contained in the database be aggregated before running the program on them.

The specification of the Dickens patent, while not specifically using the term "collectively," also clearly supports the concept of aggregating the reformatted/converted date data symbolic representations prior to further processing:

The Dickens patent specification notes:

Dates stored in symbolic form in a database are reformatted to permit easy manipulation and sorting of date-related information. (Dickens patent, Abstract)

Further the Dickens patent specification states that the claimed subject matter:

provides an approach to the representation and utilization of dates stored symbolically in databases. Existing symbolic date representations are converted to a more useful form of symbolic date representations without the addition of new data fields, and in a manner that is performed automatically by the computer and requires no user input. The approach of the invention permits direct numerical sorting of dates. (Dickens patent, Col. 1, lines 49-56)

In addition, the specification of the Dickens patent states:

In accordance with the invention, a method of processing dates stored in a database comprises the steps of providing a database with dates stored therein ... all of the dates falling within a 10-decade period of time. (Dickens patent, col. 1, lines 57-63)

Also the specification states:

The symbolic representations of the dates in the database are reformatted ... . A straightforward numerical sort of date data fields expressed in this form produces an accurate chronological ordering.

Once the symbolic representations of the dates are reformatted according to the procedures set forth above, the date information may be sorted, numeral 38, or otherwise manipulated, numeral 40, together with the entries associated with the dates. Such manipulation may include handling of data associated with the dates, storing the dates and associated information back in the data base, or other processes. (Dickens patent, col. 3, lines 39-56)

Further, the specification of the Dickens patent notes that the claimed subject matter provides:

> an effective technique for reformatting symbolic representations of date information that is rapid and automated, and yields new symbolic representations of date information that are particularly amenable to further processing. (Dickens patent, col. 3, lines 61-65)

Applicant submits that each of these references to databases would be clearly understood by those skilled in the art to relate to databases with very much more than two data entries, including at least data entries in two decades spanning the turn of the century, evincing the Y2K problem to be dealt with according to the disclosure and claimed subject matter of the Dickens patent.

Dependent claims 9 and 14 and dependent claims 10 and 15 support collectively to mean aggregating the converted symbolic representations. Claims 9 and 14 call for aggregating the converted date data symbolic representations by storing them allowing further manipulating, e.g., of the information tied to the reformatted date data symbolic representations.

Other claims added in the Reissue application and not objected to by objections upheld by the Board, other than containing the "collectively" recitation support the fact the applicant was in possession of "collectively" processing the converted date data symbolic representations. As an example, claim 58 recites "the program performs an operation which manipulates data in a data field associated with the at least one date data field ... according to the converted symbolic representation of the date." This also requires that the converted symbolic representations be aggregated before the program is run.

Applicant submits that above noted portions of the Dickens patent specification and the Board's determinations of the claimed subject matter that was in the possession of the applicant at the time of filing, support a claimed subject matter relating to a process involving converting each of the existing date data symbolic representations, aggregating the converted symbolic representations and performing further processing "collectively" on the aggregated converted symbolic representations.

Applicant further submits that this is essentially the meaning of many of the claims rejected due to the presence of the term "collectively" even with that term removed. If the Board does not agree that "collectively" is supported by the specification of the Dickens patent, as

asserted above, applicant requests that examination be reopened to allow applicant to remove the offending term from the pertinent claims.

Does Exhibit A Support the "Collectively" Recitation (Opinion, p. 70)

The Board has also taken the position that that Exhibit A does not support this interpretation of "collectively."

The program of Exhibit A performs an "extract" operation in which the date data in a tools database having a date data entry symbolic representation regarding last inventory date "last_inv" is converted for all of the entries in that field. The program indicates that the conversion function is followed by a sorting step for the converted date data symbolic representations and then followed by a printing step of the date-sorted entries. This is also followed by a "change" step where the information associated with each sorted entry is changed. Therefore the extracted entries with the changed (converted/reformatted) date data symbolic representations are aggregated and then at least sorted "collectively."

Does "Collectively" Have Multiple Plausible Meanings (Opinion, p. 71)

The Board has also taken the position that "collectively" is indefinite as having multiple plausible meanings.

Applicant submits that, as the Board has noted, there is a common meaning for "collectively" and this common meaning is supported by the specification of the Dickens patent, the claims taken as a whole and the Board's interpretation of what the disclosure of the specification of the Dickens patent would teach on skilled in the art what the inventor had possession of as of the filing of the Dickens patent application. Therefore, this meaning should be adopted, removing the possible ambiguity of different "plausible" meanings.

Further applicant submits that there are actually not two separate plausible meanings. The act of collectively further processing the converted date data symbolic representations in a collective manner or in the aggregate means the same as the further processing being performed in a collective manner. In the context of sorting or otherwise manipulating date data and associated entries from a database after the date data symbolic representations of each data entry has been converted/reformatted and aggregated, further processing of the collected/aggregated reformatted data means the same as further processing in a collective manner. The meaning of the claims is that the aggregated converted symbolic representations of the date data are sorted or

otherwise manipulated after each has been converted to the unambiguous four digit symbolic representation.

The Rejection of Claims 1-3, 5, 7, 9 and 10 Under §103(a) Over Shaughnessy and Hazama, First Subsidiary Issue (Opinion, pp. 86-89)

Applicant respectfully submits that neither Shaughnessy nor Hazama teach or suggest the claimed features of "determining a century designator C_1C_2 *for each symbolic representation of a date* ... and, reformatting the symbolic representation of the date with the values $C_1C_2Y_1Y_2$ ... *to facilitate further processing of the dates.*"

As the Board has noted Shaughnessy presents a way to solve the Y2K problem (ambiguity of data in a legacy database utilizing only two digit year designations after the turn of the century) short of converting all of the files in the database that contain such date data by "modify[ing] the computer system".[2] (Opinion, pp. 20-21, FF 12) The modification calls a subroutine whenever such a date data is encountered in a running application and converts the formats in the subroutine, performs some date operation on one or at most two converted date data and "return[s] a parameter representative of the result of the date operation to the application program ... ." (Opinion p. 21, FF 13)

As an example given in Shaughnessy if the running application needs to know if $date_1$ (contained in $field_1$ of the database) is before $date_2$ (contained in $field_2$ of the database), the two dates are sent to the subroutine where they are converted by a process that will be described herein by the shorthand "windowing algorithm."[3] Such an algorithm, utilizing a window (which may be a ten decade window) spanning the turn of the century and comparing two digit year date data to a selected year in the window (the Y_AY_B) of claim 1, a so-called "pivot year." The comparison results in the selection between possible century designators for reformatting into four digit date data. The comparison is made and the answer (yes or no) is returned to the running application indication in the form of a "parameter" that indicates whether or not $date_1$ is before $date_2$.

---

[2] Or, as is noted later in Shaughnessy, as cited by the Board "[a]ll that is required is a modification of the application program to include a plurality of different subroutines ... ."

[3] The term "windowing algorithm" for converting two digit year date data to four digit year date data has been known since at least the days of COBOL, as described in an IBM discussion of COBOL apparently originally copyrighted in 1992. http:// publib.boulder.ibm.com/infocenter/iadthelp/v6r0/index.jsp?topic= /com.ibm.etools.iseries.pgmgd.doc/c092540561.htm

This is illustrated in FIG. 8 of Shaughnessy.[4]

As the portions of Shaughnessy cited by the Board indicate Shaughnessy determines the starting date of the window without regard for what dates may be in the database. [FF 15-19] The Board has indicated that applicant's distinction of Shaughnessy for not selecting the beginning of the ten decade cycle before the earliest date in the database as missing the mark. The Board cites FIG. 4 of Shaughnessy as teaching the determination in Shaughnessy of the end of the 100 year cycle, which "necessarily determines the beginning." The Board also makes reference to Shaughnessy's use of the "number of future dating years *required.*" Applicant respectfully submits that neither of these criteria take into account, whatsoever, the earliest date in the database.

Whether Hazama discloses this aspect of the claimed subject matter and the combination of Shaughnessy and Hazama renders the claimed subject matter obvious or the Examiner properly relied on this as being simply a matter of common sense are separate issues. However, Shaughnessy actually teaches away from setting the beginning of the window according to the earliest date data in the database. Shaughnessy selects the pivot year as the year of current date or the installation date without consideration of the spread of the years in the database.

The §103 rejection of claims 1-3, 5, 7, 9 and 10 over Shaughnessy and Hazama, Second and Third Subsidiary Issues (Opinion, pp. 89-90)

Neither Hazama nor Shaughnessy teach or suggest the claimed conversion of *each* of the date data entries with the ambiguous two digit date data symbolic representation to the non-ambiguous four digit date data symbolic representation for purposes of facilitating further processing such as running a program on the so converted date data entries, such as, sorting them. The Board seems to agree with this as it reads the claimed conversion of "each symbolic representation of a date in the database" on Shaughnessy by assuming there are only two date data entries in the database.[5] The Board has taken the position that:

> This language does not place any restriction on the number of symbolic representations of dates stored in the database, except perhaps, it may be argued that more than one

---

[4] "FIG. 8 is a flowchart illustrating the steps the modified computer system performs to compare two date fields." A similar program flow chart is illustrated in FIG. 10 of Shaughnessy relating to comparing two dates encountered in the application program to determine the number of days between the two dates and returning to the application program by the called routine the number of days.

[5] Hazama similarly runs a subroutine upon encountering a date data with only two year digits and uses a windowing algorithm on the encountered date data each time one is encountered. Hazama does suggest using a windowing algorithm with a ten decade window that starts with a date no later than the earliest date in the database.

symbolic representation is implied by the use of the plural word 'dates.' Appellant does not point to anything in the Specification that requires us to interpret claim 1 to preclude the database having just two entries stored therein.

Applicant respectfully submits that the Specification and the art clearly indicate that the databases in question must be interpreted to have many more than two data entries in them, or else they would not be subject to the burdensome process of redoing the database to eliminate the Y2K problem. This was well understood by the art at the time of the filing of the Dickens patent application and clearly the subject of the solution set out in the processes of the claimed subject matter.

Applicant submits that those skilled in the art at the time of the filing of the application resulting in the Dickens patent would have understood from the disclosure thereof (with or without Exhibit A) that the databases needing the solution to the Y2K problem addressed by the claimed subject matter were existing legacy databases with thousands or tens of thousands or hundreds of thousands or millions of data entries. Clearly a database with only two date data entries would easily be addressed with a change of the data entries in the database itself, an option not economically available for the much larger databases to which the claimed subject matter is directed.

As noted below the Dickens patent expressly refers to databases with "large numbers" of date data entries. (Dickens patent, col. 2, lines 24-26) The Dickens patent and the art recognized that the databases in question, subject to "burdensome" redoing of the databases in question because of the effects of the Y2K problem were massive databases having many more than just two date data entries.

The Dickens patent notes:

> Dates stored in symbolic form in a database are reformatted to permit easy manipulation and sorting of date-related information. ... The reformatted date information is particularly useful when the reformatting is in C_1 C_2 Y_1 Y_2 M_1 M_2 D_1 D_2 format, because sorting by date is accomplished using a pure numerical-value sort. (Abstract)

Further the Dickens patent notes in regard to the scope of the claimed subject matter regarding the data bases in need of the solution to the Y2K problem addressed by the claimed subject matter and how that problem is addressed by the claimed subject matter:

[t]he present invention thus provides an efficient approach to converting and utilizing symbolic date representations in databases, which allows automatic processing of dates ranging from before to after the year 2000. *The large number of dates represented in some databases may thereby be readily processed and utilized.* (Dickens patent, col. 2, lines 22-27)

Such a database is referenced in Exhibit A. Those skilled in the art would understand that a company wide database relating to inventorying of tools would contain many hundreds if not thousands or more of date data entries relating to tools to be checked, according to the program, for a last inventory date. Thus after the conversion of the date data entries from the ambiguous two digit year designator to a four digit year designator there would be many entries to be, e.g., date sorted and manipulated in additional ways according to the program of Exhibit A.

It is clear that the Dickens patent deals with the Y2K problem. The Dickens specification notes:

However, with the turn of the century at Jan. 1, 2000, the representation and utilization of dates becomes more complex. Using the numerical form above, Dec. 15, 2000 is represented as 12/15/00. If a numerical sort is performed on 12/15/93 and 12/15/00, the later date 12/15/00 sorts as the first-occurring date, an incorrect result. (Dickens patent, col. 1, lines 25-30)

In addition, notes the Dickens patent specification:

The present invention provides an effective technique for reformatting symbolic representations of date information that is rapid and automated, and yields new symbolic representations of date information that are particularly amenable to further processing.

As discussed in more detail below, the art cited by Examiner Amsbury, who originally allowed the claims in the Dickens patent, makes it clear that the Examiner understood that the claimed subject matter was aimed at databases containing huge numbers of entries, which engendered the Y2K problem. Articles cited by the Examiner in the prosecution of the Dickens patent also indicate that the Y2K problem addressed by the Dickens patent relates to databases with many more than two date data entries using the ambiguous two digit year date designators.[6]

[6] The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation, Third Edition, May, 1996; IBM: The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation; First Edition, Oct. 1995.

The Soeder patent, 5644762, cited by Examiner Amsbury notes:

> While one solution would be to extend the date fields in databases from six bytes to eight, such a solution would require a burdensome process of rewriting much information already stored. (Soeder Patent, 5644762, Col. 1, lines 43-46)

Further notes the Soeder patent, regarding the magnitude of the Y2K problem, requiring a solution short of the "burdensome process" of reentering date data in four digit format to replace the existing two digit format:[7]

> Examples of uses for the invention include the following. In a payment processing system at a bank, it is crucial to distinguish a payment due date in 2000 from one in 1900. Even commercial computer systems that do not process payments, such as airline reservation systems, frequently process date-sensitive information.[8] (Soeder Patent, 5644762, Col. 3, lines 45-50)

Another patent cited by the Examiner Amsbury was the patent to Mao, 5668989. The Mao patent also recognizes that the Y2K problem related to databases with many more than 2 date data entries.[9] Articles referenced in Mao also all deal with the Y2K as it relates to huge databases difficult to change to four digit year date data from the existing two digit year date data.[10]

The Mao patent notes:

> Most existing application software in business data processing area treat a date in a format similar to "MM/DD/YY" or "YY/DDD", using 2 digits to carry the last 2 digits of a 4 digit year number, resulting in 2 digit year numbers. ... This problem is known as the "Year 2000 (Y2K) Problem" in the industry, and has been considered a crisis.

---

[7] An article cited in the prosecution of the Soeder patent indicates the types of databases to which the Dickens and Soeder approaches to the Y2K problem relate. Hart et al, "A Scaleable, Automated Process for Year 2000 System Correction", PROC 18th International Conference on Software Engineering, 25-30 Mar. 1996, pp. 475-484. Mar. 30, 1996.

[8] The Soeder patent proposed using dates in integer format for dates after December 31, 1999.

[9] The Mao patent proposes using "biased 2 digit 'hybrid radix' numeric fields for inputting, generating, storing, processing, and outputting year numbers ranging from 1900 to 2059 in a data processing system."

[10] Zvegintzov, N. "The Year 2000 as racket and Ruse." American Programmer, Feb. 1996, vol. 9, No. 2, pp. 16-19. . Hart, et al. "A Scaleable, Automated Process for Year 2000 System Correction", Software Engineering Int'l Conf. 1996, pp. 475-484; McCabe, T. "Cyclomatic Complexity and the Year 2000" IEEE Software, vol. 13, Issue 3, May 1996, pp. 115-117; Glass, R. L. "The Next Date Crisis and the Ones After That" Communications of the ACM, vol. 40, No. 1, Jan. 1997, pp. 15-17; Jerome T. Murray et al.: "The Year 2000 Computing Crisis", McGraw-Hill, 1996. . http://www.software.ibm.com/year2000/faq.html, "Frequently asked questions about the year 2000 challenge", IBM 1995; http://www.storage.ibm.com/storage/software/sort/srtmycw.htm, IBM, 1995 "Set the Century window for two-digit years"; http://www.storage.ibm.com/storage/software/sort/srtmytr.htm, IBM, 1995 "Transform two-digit year dates to four-digit year dates".

It is clear that Examiner Amsbury in originally allowing the claims in the Dickens patent also did not view the databases as to which the claimed subject matter applied as containing one or two data entries. Otherwise Shaughnessy would have been applied to the claims, which Examiner Amsbury did not do.

Other aspects of the prosecution of the Dickens patent application, such as swearing behind the IBM paper regarding solutions to the Y2K problem indicate that Examiner Amsbury was well aware of the meaning of "database" as used in the specification and claims of the Dickens patent.

Clearly those skilled in the art would understand the databases in question to contain huge amounts of data entries including date data entries and perhaps even more than two date data fields, as opposed to no more than two date data entries.[11] Applicant submits that it would have been clear to one skilled in the art that such "databases" as claimed potentially contain huge numbers of date data entries, thousands to millions of date data entries. Such databases, of the type containing "large numbers of dates," as noted in the Dickens patent specification, are the databases to which the claims refer. A database with two or less date data entries, if one such actually exists anywhere, would not need the Y2K solution of the Dickens patent. In fact, unless the database is one which has no date data field at all (and thus, also no Y2K problem), the fact the database had two or less entries would negate the need for any "database" per se.

Applicant, therefore, respectfully submits it is not correct to interpret the claim to read on a "database" that includes only two date data entries, and, therefore, read the claims on the process of Shaughnessy.

The Board has taken the position that Shaughnessy teaches:

> reformatting the symbolic representation of the date to yield a CCYYMMDD date format and returning a date field with the converted date to the subroutine.

Applicant respectfully submits that Shaughnessy does not return any reformatted symbolic representation to the running application that called the date processing subroutine. Rather it returns a "parameter representative of the result of the date operation ... ." (FF 13) As noted above, this "parameter" is in the nature of the answer to a question, such as, which of two

---

[11] See also, Ohms, p. 249:

> When millions of dates are stored, as they are in most business systems, every additional byte required to save a single date multiplies to millions of additional bytes of storage. The programming necessary to accommodate larger date-field sizes in records further complicates date conversion.

dates encountered while running the pertinent application program that called the subroutine is the older or whether or not an encountered date is before or after some reference date. This is not the "return[of] a date data field with the converted date", as indicated by the Board.

Therefore, the support for affirming the Examiner as to subsidiary issues two and three (Opinion, pp. 89-91) is not entirely well founded.

The §103 rejection of claims 1-3, 5, 7, 9 and 10 over Shaughnessy and Hazama, Fourth Subsidiary Issue (Opinion, pp. 41-43)

The Board has taken the position that applicant:

> has not convincingly explained why Shaughnessy teaches away or why Shaughnessy's method of operation would need to be changed in order to incorporate the teachings of Hazama.

Applicant respectfully submits that both Shaughnessy and Hazama teach away and would have to be significantly modified in theory of operation to teach or suggest the claimed subject matter of the Dickens patent.

Shaughnessy cannot perform the claimed process, and therefore, teaches away. Shaughnessy by returning a "parameter" to the program cannot reformat each or all of the date data representations in the data base and then perform further programming "sorting" or "manipulating" on the reformatted date data, since the "parameter" returned to the program is specific to an operation, e.g., comparison, specific only to the two particular date representations being operated on by Shaughnessy for purposes of returning the parameter to the program. The "parameter" simply indicates, e.g., the a date is before or after a reference date or one date is greater than, equal to or less than the other, and is not correlated to any other date data representation in the database for purposes of further processing. It is, therefore, not a conversion and reformatting that "facilitates further processing of the dates" taken from the database as claimed.

The "reformatting the symbolic representation of the date" must be read in the context of the earlier claim recitations that define the context of this process limitation to mean performing reformatting on "each" or "all" date data representations that have previously been the subject of the claimed form of use of a windowing algorithm prior to this reformatting step. Further the claimed reformatting is to "facilitate further processing of the dates." As noted above, Shaughnessy reformats at most two pieces of date data information from the database at a time

and returns a parameter that gives a result of processing done in the called up subroutine of Shaughnessy, but does not "facilitate further processing of the dates."

Shaughnessy refers to "passing at least one date field *to* the subroutine [and] a means for *returning a parameter* to the application ... . " The "parameter," is not a reformatted date or dates, and is "for use by the application program in further operation." (Shaughnessy, col. 2, lines 53-54, emphasis added) An example of this is "[i]f the result received from the subroutine [the parameter] indicates that the date the next payment is due is greater than today's date [the program can go on to] indicate that the account is OK." (Shaughnessy, col. 4, lines 59-61) Shaughnessy also notes as an alternative to the above, the Shaughnessy process may "pass[] at least one date field which is representative of at least two dates *to* the subroutine, determining which of the two dates corresponds to the date field operation according to a predetermined criteria, performing the date operation on the date field, and *returning the parameter* to the application program ... ." (Shaughnessy, col. 2, lines 59 - 64, emphasis added).

Thus Shaughnessy, and also Hazama, teaches running an application program, encountering a date data entry while running the program, halting the program to call a subroutine to convert the date data and, in the case of Shaughnessy, returning to the running application program a "parameter" indicating the result of some date comparison operation on the single date data entry (or at most two date data entries).[12] This is distinct from and teaches away from the claimed subject matter, where each symbolic representation of date data in ambiguous two digit format is converted to unambiguous Y2K four digit format, followed by sorting or other manipulation of this body of (aggregation of, mass of) converted date data symbolic representations. The calling of the subroutine in Shaughnessy and Hazama responsive to encountering a lone date data entry (or two date date entries to be compared) is at least very slow compared to the claimed subject matter, and at worst will not allow for certain programs to even effectively run on the data in the database. For example, date sorting of all of the date data entries in the database is not possible when the called routine can only see two date data entries at a time, as called for by the process of Shaughnessy.

[12] Hazama is not clear exactly what is returned to the running application from the subroutine. However, even if expressly or by implication the converted date data representation is returned to the running application, this one at a time subroutine calls based on encountering a date data bears the same inefficiencies as Shaughnessy and is likewise distinguishable over the claimed subject matter.

The Rejection of Claims 1-3 under 35 U.S.C. §103(a) Over Ohms in View of Hazama First, Subsidiary Issue (Opinion pp. 93-97)

The Board has taken the position with regard to the first subsidiary issue regarding the rejection of claims 1-3 under 35 U.S.C. §103 (a) over Ohms in view of Hazama that "[t]he teachings of Ohms include and apply to dates in Gregorian format [and] appellant has misread the teachings of Ohms." (Opinion, p. 97) Applicant respectfully submits Ohms' teaching about Gregorian (or Julian) dates does not apply to date data actually stored in the database. As noted in FF 41, Ohms states that:

> [t]he two positions traditionally used in both Julian and Gregorian date formats[13] implicitly represent a year within a century [but] *this system is inadequate for representing dates in more than on century* [but] the *Lilian date* [actually used by Ohms to store dates in the database, (see FF 40)] *avoids the ambiguity by using seven positions* [a seven digit binary number from 1 to 2^7] *for the number of days from the beginning of the Gregorian calendar.*

Ohms notes "[t]he use of a format termed Lilian date format ... is introduced.

As discussed in FF 42-43, the reformatting done by Ohms is only for convenience in data entry using two digit year date data format. The actual data stored in the database and manipulated is in Lilian format which does not even include year designation digits.

Ohms notes:

> [e]nd users [during data entry] usually enter two digits for the year ... . [T]o avoid adverse user reaction programs [that enter data into the database] must continue to function with only two digits for year.

In the Abstract of Ohms partially quoted in FF 40, Ohms is clearly talking about "existing date formats" as the Lilian format that can be used to solve the Y2K problem. Ohms notes that "[t]he use of existing date formats can eliminate the need for massive system modifications." The massive system modifications are explained further to be caused by the ambiguity across the century boundary of dates stored in Gregorian or Julian form with only two digit year date designators which are "inadequate for representing dates in more than one century." (FF 41)

Ohms also notes:

---

[13] The difference between the two is that the day of the year in Gregorian is in mm/dd format and in an integer format in Julian (days 1-365, except in leap years), and when the starting date of the respective calendar. Thus each contain year designators which, if stored in a database in two digit format, evoke the Y2K problem.

The Gregorian calendar serves us quite well in our day-to-day living. Due to discontinuities in various date divisions, however, it is not readily adaptable to computer programming. This fact becomes more apparent as we approach the new century. (Ohms, p. 244)

Further Ohms states:

[a] format termed the Lilian date format presented here as the basis for making date conversions of the type mentioned earlier. The Lilian format, which may be stored in three bytes of memory, provides the storage capacity for dates well beyond the year 10000. This format handles processing across century years and other aspects of date conversion not currently adaptable to computer programming. (Ohms, pp. 244-245)

Ohms further explains that this is accomplished by using the Lilian format which (like Booth's integer format) avoids the two digit date designator ambiguity problem (the Y2K problem) because the seven digit Lilian date number is decoded unambiguously to a particular year associated with each Lilian formatted date number stored.

And example of this is given in Ohms:

Lilian to extended Julian. The purpose of this example is to show the procedure for converting a Lilian date to an extended Julian date. First, create a virtual Lilian date, with a starting point of January 1, 1201. Convert this result into a Julian calendar (not Julian format) date. ... The result is the number of full Julian calendar days from January 1, 1201, to the date being converted. We now convert this to an extended Julian format date. (Ohms, p. 247-48)

Ohms then goes on to explain a windowing algorithm for use in "avoiding adverse user reaction" from users that desire to continue the entry of data in two digit year format. (Ohms, p. 248)

In summation Ohms refers to the use of Lilian to avoid the limits of the traditional Gregorian (or Julian) date formats with two digit date designators while allowing for use of two digit date designators for user interface with the computer (data entry and/or display}, as follows:

Programmers require date-processing functions that effectively handle applications for both the present and the future. ... For the future, additional functions are required that support processing restricted only by the limitations of the Gregorian calendar. These functions must be fully compatible with existing date-format and record-size restrictions.

> Massive conversion efforts should not be required to process and store dates outside the twentieth century [by storing and processing date data in Lilian format]. Also, *end users should be able to continue to use existing two-digit date formats when interacting with computer systems.* (Ohms, p. 250)

In other words, Ohms teaches not to store dates in the database in Gregorian (or Julian) format with two date data year digits, because they are ambiguous across the turn of the century, but instead store them in Lilian (equivalent to the integer dates taught by Booth, except for the start date of the integer count). The Lilian format for date data stores a unique number for each day, which unique number includes the year in which the day occurs. Ohms also teaches, however, to use a windowing algorithm to allow data entry to continue to use only the two digit date data format (a "typical" format for Gregorian or Julian dates).

Booth and Ohms describe systems that employ database date data formats that are not ambiguous due to the occurrence of the turn of the century. Both Lillian date data format[14] and integer number date data format fully define the date stored in such format and do not need to be further converted for such processing as sorting and the like. (Opinion, p. 34, FF31 and pp. 38-FF39, FF41) Nothing in Booth indicates that manipulation of dates stored in the database in integer format needs to convert the stored format in any way. The portions cited by the Board make this clear. (Opinion, p. 34, FF31) Applicant submits that "[a]dding an integer to a date will result in a future date," and "[s]ubtracting two dates will result in a number of days between the two," means the dates are stored and manipulated in integer date data format and cannot be so manipulated if stored in a year/month/day type of format, whether the format is the ambiguous YY for or the unambiguous YYYY format.[15]

The same is true of Ohms' use of Lillian number format for actual data storage in the database and manipulation of the data. (FF 41)

Booth and Ohms each describe the use of a windowing algorithm, but not in the environment of the claimed subject matter ("a database with symbolic representations of dates stored therein according to a format wherein ... Y_1Y_2 is the numerical year designator" and not for the purpose of "facilitat[ing] further processing of the dates." Booth uses a windowing

---

[14] Lillian date format is a form of integer date data format based on the Gregorian calendar.

[15] Applicant submits that the discussion of the Booth SORT function does not deal with any conversion of the format of the date data being sorted, before or during the sort and is not inconsistent with applicant's interpretation of the teaching of Booth.

algorithm for date data entry and display but does not use it to reformat date data stored in the database in the ambiguous two digit date data format to facilitate further processing, because the date data is not stored in Booth in the ambiguous two date digit format. Ohms uses a windowing algorithm to enable date data entry in two digit year date data format, but also actually stores the date data and manipulates the date data in unambiguous Lilian date format.

The Rejection of Claims 1-3 under 35 U.S.C. §103(a) Over Ohms In View of Hazama Second and Fourth Subsidiary Issues (Opinion, pp. 97-99)

The Board has taken the position with regard to the second and fourth subsidiary issues in regard to the rejection of claims 1-3 under 35 U.S.C. §103 (a) over Ohms in view of Hazama that "the teachings of Ohms are not limited to dates in Lilian format and also include dates in Gregorian format."

Applicant respectfully submits that FF 41, as discussed above, clearly indicates that, insofar as dates stored in and manipulated in the format stored, Gregorian date formats are replaced by Lilian date format by Ohms to avoid the ambiguity at the turn of the century arising from a two digit date data format. The portion of Ohms quoted by the Board in regard to the fourth subsidiary issue, as noted above, is contrary to interpreting Ohms as storing dates in a database with two or four digit date data representations, as opposed to Lilian format without any year date data representations.[16] The Lilian date data representation format taught by Ohms simply has no YY or YYYY digits and Ohms teaches not to use such formats in the database, but rather to use a Lilian format.

Accordingly, applicant respectfully submits that Ohms does not teach or suggest a database *with date data stored with symbolic representations including two digit date symbolic representations* (first subsidiary issue), nor *determining a century designator for dates stored in the database* with the two digit date symbolic representations (fourth subsidiary issue), nor *reformatting the dates stored in the database* to facilitate further processing (fifth subsidiary issue). Hazama does not teach or suggest determining a century designator for each of the dates or reformatting each of the dates *to facilitate further processing*. Ohms is, for the reasons noted above, not analogous art for the claimed subject matter relating to converting dates in databases to remove the Y2K ambiguity to facilitate further processing, since Ohms does not store dates in

[16] Of course there is a date designator in a Lilian formatted date, but it is embedded in the unique number for any given day in a sequence of days from the start date of the calendar each represented by successive numbers in a sequence of numbers.

the database using the ambiguous two digit date symbolic representations, and in fact teaches not to use such format and does not convert those stored date data symbolic representations to the unambiguous format to facilitate further processing. Ohms stores and manipulates the date data in Lilian format.

The Rejection of Claim 5 under 35 U.S.C. §103(a) over Shaughnessy in view of Hazama (Opinion, p. 101-102)

Applicant submits that, for the reasons discussed above regarding claim 1, neither Shaughnessy nor Hazama, nor the combination of the two teaches or suggests the claimed "reformatting *each* symbolic representation" (claim 5) for the purpose of "facilitating further processing" (claim 1). Claim 5 also depends from allowable claim 1 and should be allowed for that reason.

The Rejection of Claim 5 under 35 U.S.C. §103(a) over Ohms in View of Hazama (Opinion, p. 102)

Applicant submits that, for the reasons discussed above regarding claim 1, neither Ohms nor Hazama, nor the combination of the two teaches or suggests the claimed "reformatting *each* symbolic representation of a date [*in the database*]" (claim 5) for the purpose of "facilitating further processing [*of the reformatted symbolic representations*]" (claim 1). Claim 5 also depends from allowable claim 1 and should be allowed for that reason.

The Rejection of Claim 9 under 35 U.S.C. §103(a) over Shaughnessy in view of Hazama (Opinion, pp. 103-104)

The Board has taken the position that Shaughnessy teaches "storing the symbolic representations of dates and their associated information back into the database." Applicants submit that Shaughnessy teaches "convert[ing] all dates within the application system of the computer to use date fields with four digit representations for the year." Shaughnessy is not talking here about the particular form of using the windowing algorithm later described as being used in the called subroutine of Shaughnessy. Shaughnessy is talking about changing the existing database to remove the date field with ambiguous two digit date year data and replacing that field with a new field having the unambiguous four digit date year data. This process, as recognized by applicant's specification as well as Shaughnessy

Applicants further submit that, for the reasons noted above regarding claim 1, this is not the claimed converting of each date in the database from a two digit year designator format to a

four digit year designator format for purposes of further processing of the reformatted dates and their related data fields. The claimed subject matter avoids the need to "convert [the existing date data fields in the database] to include date fields having three or four digit representation for the year." This process, as indicated by Shaughnessy would be "expensive and time-consuming" not to mention fraught with possible error.

Claim 9 depends from allowable claim 1 and should also be allowed for that reason.

The Rejection of Claim 9 under 35 U.S.C. §103(a) over Ohms in View of Hazama (Opinion, pp. 104-105)

For the reasons noted above regarding claim 1, Ohms teaches storing date data in the database in Lilian format and does not teach reformatting the Lilian format, so it also does not teach the claimed "storing the [reformatted] symbolic representations of dates ... into the database."

Claim 9 depends from allowable claim 1 and should also be allowed for that reason.

The Rejection of Claim 10 under 35 U.S.C. §103(a) over Shaughnessy in view of Hazama (Opinion, p. 105)

For the reasons noted above regarding claims 1 and 9, Shaughnessy does not reformatting each of the dates in the existing two digit legacy database field to a four digit date data symbolic representation, then storing the new symbolic representations back in the database, then manipulating based on the new format. This necessarily at least implies storage in a newly established date data field, since the existing field set up for two digit year designation would not operate correctly if four digit date year designators were attempted by be inserted where the existing two digit data once was. This is, therefore, not the suggested solution of Shaughnessy of changing the existing date data field and reentering the data in four digit format rather than two digit format.

Claim 10 depends from allowable claim 1 and should also be allowed for that reason.

The Rejection of Claim 10 under 35 U.S.C. §103(a) over Ohms in view of Hazama (Opinion, p. 105-106)

For the reasons noted above regarding claims 1 and 9, Ohms teaches storing date data in the database in Lilian format and does not teach reformatting the Lilian format, so it also does not teach the claimed "manipulating information in the database having the reformatted date information therein."

Claim 10 depends from allowable claim 1 and should also be allowed for that reason.

The Rejections of Claims 4 and 6 under 35 U.S.C. §103(a) over Shaughnessy in view of Hazama and Booth First Subsidiary Issue (Opinion, p. 106-108)

Applicant respectfully submits that for the reasons noted above with regard to claim 1 distinguishing over Ohms the noted portions of Booth are not for sorting of dates contained in the database, but for date data entry (FF 34, "when a two-digit year is entered", FF 35), extraction (FF 36, converting stored dates from integer to string formats), or display (FF32 "control the display of dates"). Booth (FF 31) teaches storing and manipulating date data in the database in integer format - each integer representing a specific unique day (or even hour or minute) in time, which necessarily is not year ambiguous.

Booth and Ohms describe systems that employ database date data formats that are not ambiguous due to the occurrence of the turn of the century. Both Lillian date data format[17] and integer number date data format fully define the date stored in such format and do not need to be further converted for such processing as sorting and the like. (Opinion, p. 34, FF31 and pp. 38-FF39, FF41) Nothing in Booth indicates that manipulation of dates stored in the database in integer format needs to convert the stored format in any way. The portions cited by the Board make this clear. (Opinion, p. 34, FF31) Applicant submits that "[a]dding an integer to a date will result in a future date," and "[s]ubtracting two dates will result in a number of days between the two," means the dates are stored and manipulated in integer date data format and cannot be so manipulated if stored in a year/month/day type of format, whether the format is the ambiguous YY for or the unambiguous YYYY format.[18]

The same is true of Ohms' use of Lillian number format for actual data storage in the database and manipulation of the data. (FF 41)

Booth and Ohms each describe the use of a windowing algorithm, but not in the environment of the claimed subject matter ("a database with symbolic representations of dates stored therein according to a format wherein ... Y_1Y_2 is the numerical year designator" and not for the purpose of "facilitat[ing] further processing of the dates." Booth uses a windowing algorithm for date data entry and display but does not use it to reformat date data stored in the

[17] Lillian date format is a form of integer date data format based on the Gregorian calendar.

[18] Applicant submits that the discussion of the Booth SORT function does not deal with any conversion of the format of the date data being sorted, before or during the sort and is not inconsistent with applicant's interpretation of the teaching of Booth.

database in the ambiguous two digit date data format to facilitate further processing, because the date data is not stored in Booth in the ambiguous two date digit format. Ohms uses a windowing algorithm to enable date data entry in two digit year date data format, but also actually stores the date data and manipulates the date data in unambiguous Lilian date format.

The Board has taken the position that the claim 4 does not preclude the dates being in an integer format. Applicant respectfully submits that the claims call for data stored in the database in the existing date data format in the database to be a $Y_1Y_2M_1M_2D_1D_2$ format. This is clearly not integer format as that term is used in the art, including Booth.

Claims 4 and 6 depend from allowable claim 1 and should be allowed for that reason.

The Rejections of Claims 4 and 6 under 35 U.S.C. §103(a) over Shaughnessy in view of Hazama and Booth Second Subsidiary Issue (Opinion, p. 108-109)

The Board has taken the position that:

> Booths' teaching of sorting reformatted dates would facilitate Shaughnessy-Hazama's system to return reformatted dates in chronological sequence.

Applicant respectfully submits that Booth teaches arranging reformatted dates, reformatted from the non-year ambiguous integer form in which they are stored and manipulated into string format, such as $Y_1Y_2Y_3Y_4M_1M_2D_1D_2$ for human interface, and putting those, e.g., in chronological order. (FF 36)

The Board has taken the position that:

> the two digit year format (FF 33) disclosed by Booth is subject to theY2K ambiguity problem just as with Shaughnessy and Hazama. Appellant has not convincingly explained why Shaughnessy's and/or Hazama's principle of operation would need to be changes to incorporate the sort teachings of Booth or why these references would be rendered inoperative if the sort teachings of Booth were incorporated.

Applicant respectfully submits that the cited portion of Booth has nothing to do with the Y2K problem. (FF 33, See also FF 36) Booth does not have a Y2K problem as to dates stored in and processed in the database, including, e.g., sorting, with such stored date data in integer format. (FF 31) In converting a date in integer format to a date in string format (whether with two digit or four digit year designators) the conversion is completely unambiguous. Each integer stands for a unique and specific day (or perhaps an hour or hour and minute of a unique and

specific day). The integer date already includes the year in which that day of the date is contained, so the integer format is completely unambiguous even across a century boundary.

When Booth talks about the "SET DATE FORMAT" command Booth is talking about setting the date format for user interface purposes. The data in the database remains in integer format and is processed in such format. When Booth talks about "convert[ing] dates to string format" he is referring to converting the dates from integer format as stored and manipulated to string format which stored dates, and therefore which conversion, has no need for the Y2K solutions proposed by Shaughnessy, Hazama or applicant.

One skilled in the art would understand that having a database as Booth describes the date data is already in a completely and utterly unambiguous format vis a vis in what century the date is (i.e., having no Y2K problem whatsoever). It would be foolish and wasteful programming to convert this data from the integer format into a string format such as $Y_1Y_2M_1M_2D_1D_2$, which has the century ambiguity, and then use Shaughnessy or Hazama or the combination of the two to reconvert the data into an unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ format to facilitate further processing.

It would also be foolish and wasteful programming to convert the integer data to string data directly into the unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ format to facilitate further processing. The dates in Booth are already in a format with "powerful date manipulation capabilities." (FF 31) In this regard Booth notes:

> [d]ates are stored *internally* [i.e., in the database] in such a way [*integer format*] that math operations can be performed on them to derive other dates, [for example] [a]dding an integer to a date will result in a future date, [and] [s]ubtracting two dates will result in the number of days between the two.[19]

These operations cannot be done for dates in a year/month/day format, even the unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ format. As an example adding 25 to a representative date such a 19991210 (December 10, 1999) gives the non-existent date of 19991235 (December 35, 1999) and subtracting 25 from the same representative date gives the non-existent date of 19991185 (November 85, 1999).

---

[19] Booth is talking here about a form of integer date where each successive integer in a sequence represents a new day. Further, of course, sorting is simply a numerical sort.

Therefore, there is no need for a Y2K solution for Booth's database, rendering the solutions of Shaughnessy, Hazama, applicant or the art cited in the prosecution of applicants patent application superfluous, unusable, wasteful and needles programming, and thus not subject to being combined as references, because Booth teaches away from using a database with ambiguous string date data formats such as either of the unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ or ambiguous $Y_1Y_2\,M_1M_2D_1D_2$ formats. In addition, the solution proposed by Shaughnessy and/or Hazama for date data stored in the ambiguous $Y_1Y_2\,M_1M_2D_1D_2$ format would not work with data as stored in the Booth database, unless the programmer took the foolish and wasteful programming steps of converting unambiguous integer date data representations into the ambiguous $Y_1Y_2\,M_1M_2D_1D_2$ string format solely for the purpose of having the Y2K solutions proposed by Shaughnessy or Hazama or the combination of the two be useful. This is also even more unlikely a choice by one skilled in the art, since the integer date data of the Booth database is easily convertible to the unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ string format, also avoiding any need for any Y2K solution, including those proposed by Shaughnessy or Hazama or the combination of the two.

The Rejection of Claim 6 under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama and Booth (Opinion p. 110)

Applicant respectfully submits that Booth teaches, at best, sorting symbolic representations of dates reformatted from unambiguous integer date data format into unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ format rather than the claimed date data residing in the database in the ambiguous $Y_1Y_2\,M_1M_2D_1D_2$ string format that has been converted to the unambiguous $Y_1Y_2\,Y_3Y_4M_1M_2D_1D_2$ string format.

The Rejections of Claims 11-15 under 35 U.S.C. §103(a) (Opinion, p. 113)

Applicant respectfully submits that claims 11-15 are method claims containing recitations essentially the same as a related claim in claims 1-10, and for the reasons argued above are allowable over the references cited.

The Rejection of Claim 68 under 35 U.S.C. §103(a) over Shaughnessy in View of Hazama and Ohms in View of Hazama (Opinion pp. 113-114)

Claim 68, like claims 1 and 4 contains the recitations of "determining a century designator for each symbolic representation of a date in the database" and reformatting the symbolic representations of each symbolic representation of a date ... in the database" and "to

facilitate further processing of the reformatted symbolic representations." For the reassons discussed above with regard to claims 1 and 4, neither Shaughnessy nor Hazama, nor the combinations of the two teaches or suggests these claim features.

In addition, claim 68 recites "reformatting ... without the addition of any new data field to the database" and "further processing ... by running a program on the reformatted symbolic representations ... ."[20]

Applicant respectfully submits that these added recitations of claim 68 even further distinguish the process claimed from that shown in Shaughnessy or Hazama of the combination of the two. Neither Shaughnessy nor Hazama reformat each of the symbolic representations as part of a process to facilitate further processing of the reformatted dates, so, necessarily they do not show such a reformatting step done without the addition of a new date data field. Neither teaches or suggests reformatting each of the date symbolic representations for each of the date data entries in the database to facilitate further processing involving "running a program on the reformatted symbolic representations ... ."

In addition to the arguments made above regarding the rejection of claims 1 and 4 over Ohms in View of Hazama, relating to the similar recitations in claim 68, applicant respectfully submits that since Ohms does not reformat any dates stored in the database in Y2K ambiguous format for purposes of further processing the date data with the reformatted Y2K unambiguous format, and since Ozama does not reformat each date in the database to facilitate further processing of the date data, as noted above, necessarily neither performs the additional process recitations just note with respect to claim 68.

The Rejection of Claim 69 Under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama or Ohms in View of Hazama and Booth (Opinion pp. 114-115)

Claim 69 contains the same recitations noted above with respect to claim 68 in common with claims 1 and 4, with the addition of the recitation of "running a program on the reformatted symbolic representations ... ."[21] For the reasons just noted regarding claim 68 neither Shaughnessy nor Hazama nor the combination of the two teach or suggest running a program on the reformatted date data symbolic representations after the reformatting of each one in the

---

[20] Claim 68 does not contain the recitations "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator". These have not been argued to distinguish over the cited art.

[21] Claim 69 does not contain the recitations "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator". These have not been argued to distinguish over the cited art.

database. Similarly this is not taught or suggested by Ohms in view of Hazama as noted regarding claim 68. Booth adds nothing to the combination, as noted above, since Booth, like Ohms, stores data in an integer format and has no need to and does not convert data in a database from a Y2K ambiguous format to a Y2K non-ambiguous format to facilitate further processing of the date data stored in the database, and, therefore also does not run a program on such converted date data.

The Rejection of Claim 73 Under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama or Ohms in View of Hazama (Opinion, p. 115)

Claim 73 is the same as claim 1 without the recitation "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator." Since these aspects of the date data format stored in the database are not important to the date data being in Y2K format needing conversion to a Y2K unambiguous format to facilitate further processing, these recitations are not needed to distinguish over the art, and the same arguments made above as to the rejections of claim 1 apply here.

The Rejection of Claim 74 Under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama and Booth or Over Ohms, Hazama and Booth (Opinion, p. 116)

Claim 74 is the same as claim 4 without the recitation "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator." Since these aspects of the date data format stored in the database are not important to the date data being in Y2K format needing conversion to a Y2K unambiguous format to facilitate further processing, these recitations are not needed to distinguish over the art, and the same arguments made above as to the rejections of claim 4 apply here.

The Rejection of Claim 75 Under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama and Booth or Over Ohms, Hazama and Booth (Opinion, p. 116-117)

Claim 75, like claim 68, includes the elements of claim 1[22] with the additional recitation of "without the addition of any new date data field to the database". For the reasons discussed above in regard to this aspect of claim 68 distinguishing over the art cited against claim 68, claim 75 should be allowable. In addition, as noted above with respect to the rejection of 69 over Ohms in View of Hazama and Booth, the Booth reference adds nothing to that combination or to

[22] Claim 75 does not contain the recitations "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator". These have not been argued to distinguish over the cited art.

the combination of Shaughnessy and Hazama, since Booth, like Ohms, stores data in an integer format and has no need to and does not convert data in a database from a Y2K ambiguous format to a Y2K non-ambiguous format to facilitate further processing of the date data stored in the database, and, therefore also does not run a program on such converted date data.

The Rejection of Claim 76 Under 35 U.S.C. §103(a) Over Shaughnessy in View of Hazama and Booth or Over Ohms, Hazama and Booth (Opinion, p. 117-118)

Claim 76, like claim 69, includes the elements of claim 4[23] with the additional recitation of "without the addition of any new date data field to the database". For the reasons discussed above in regard to this aspect of claim 69 distinguishing over the art cited against claim 69, claim 76 should be allowable. As noted above with respect to the rejection of 69 over Ohms in View of Hazama and Booth, the Booth reference adds nothing to the combination of Shaughnessy and Hazama, since Booth, like Ohms, stores data in an integer format and has no need to and does not convert data in a database from a Y2K ambiguous format to a Y2K non-ambiguous format to facilitate further processing of the date data stored in the database, and, therefore also does not sort such converted date data.

Conclusion

Applicant respectfully submits that applicant does not claim to be the inventor of the windowing algorithm nor its use in removing ambiguity from date data having only two year designator digits, nor sorting algorithms sorting date data. The claimed subject matter relates to a particular use of windowing in the context of enabling processing of data stored in legacy databases in the Y2K ambiguous two digit date designators format. Ohms and Booth address the Y2K problem by avoiding having dates stored in their respective databases in a Y2K ambiguous format. The Lilian and essentially equivalent integer formats of Ohms and Booth respectively are by definition non Y2K ambiguous. Shaughnessy and Hazama propose inefficient and ineffective calls to a subroutine each time a running program encounters a date data entry in an ambiguous format. This is very different from the claimed subject matter of reformatting all of the dates in a date data field and then further processing such as sorting or otherwise manipulating the body of reformatted date data.

---

[23] Claim 75 does not contain the recitations "M_1M_2 is the numerical month designator, D_1D_2 is the numerical day designator". These have not been argued to distinguish over the cited art.

Respectfully submitted

William C.Cray
Reg. No. 27,627
714-536-5043
Patent Counsel, Dickens-Soeder2000, LLC